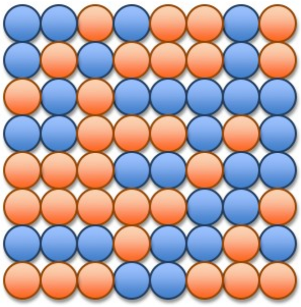


UNIT 11C

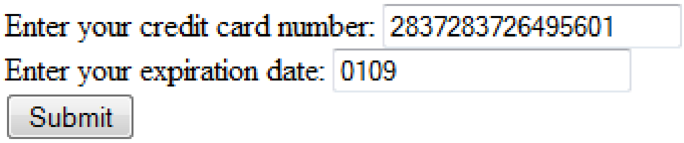
The Internet: Encryption

Data Privacy

- On the Internet and any computer network, any transmitted data can be intercepted and copied.
- Suppose you send an email to a friend.
  - Who has access to that email?
  - What if you encrypt the message so it is private?
  - Can someone who intercepts the email decrypt it?
  - Who would be against email encryption?

A Shady Example

- I want to make a purchase online and click a link that takes me to http://www.sketchystore.com/checkout.jsp
- What I see in my browser:



Enter your credit card number: 2837283726495601

Enter your expiration date: 0109

A Shady Example (cont'd)

- When I press SUBMIT, I send this:

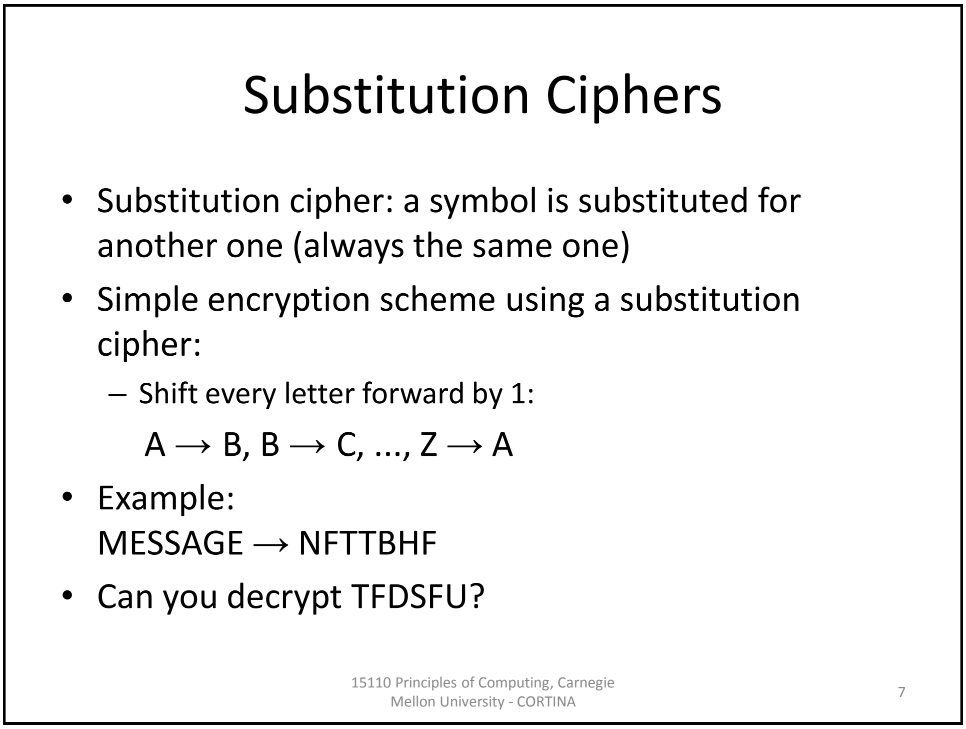
```
POST /purchase.jsp HTTP/1.1
Host: www.sketchystore.com
User-Agent: Mozilla/4.0
Content-Length: 48
Content-Type: application/x-www-form-
  urlencoded
userid=tom&creditcard=2837283726495601&
  exp=01/09
```

A Shady Example (cont’d)

- If this information is sent unencrypted, who has access to my credit card number?
  - Other people who can connect to my wireless ethernet?
  - Other people physically connected to my wired ethernet?
- When I send a letter through the mail, it passes through the hands of many mail carriers. What keeps them from reading my mail?
  - What if I send a postcard?
- Packets are passed from router to router.
  - All those routers have access to my data.

Encryption

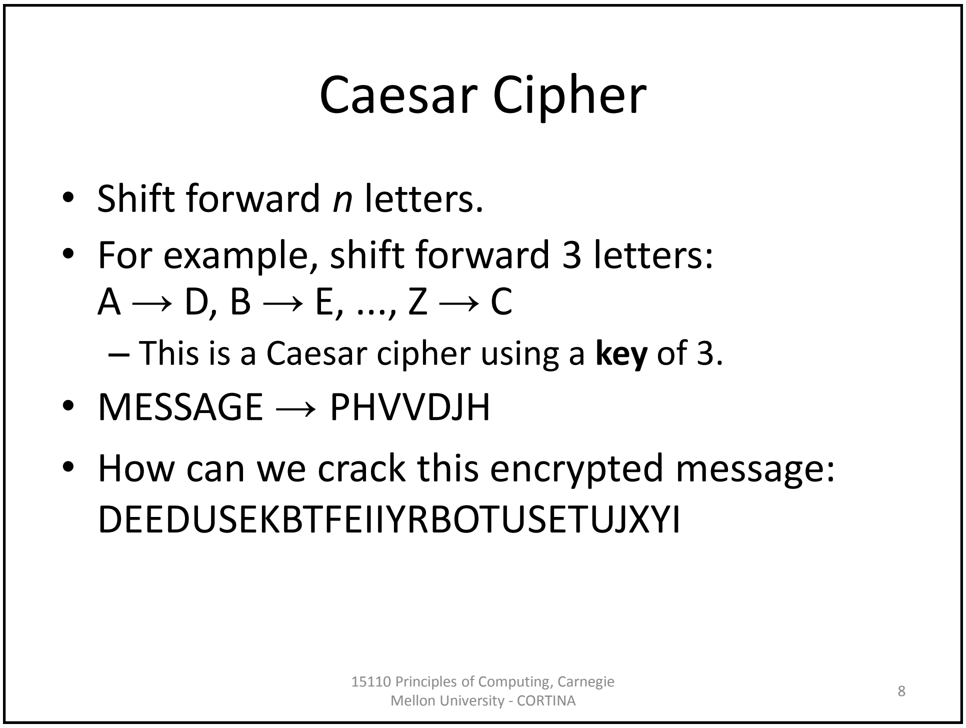
- We need to encrypt (encode) our data so others can’t understand it (easily) except for the person who is supposed to receive it.
- We call the data to encode plaintext and the encoded data the ciphertext.



Substitution Ciphers

- Substitution cipher: a symbol is substituted for another one (always the same one)
- Simple encryption scheme using a substitution cipher:
  - Shift every letter forward by 1:

    A → B, B → C, ..., Z → A
- Example:
  MESSAGE → NFTTBHF
- Can you decrypt TFDSFU?



Caesar Cipher

- Shift forward *n* letters.
- For example, shift forward 3 letters:
  A → D, B → E, ..., Z → C
  - This is a Caesar cipher using a **key** of 3.
- MESSAGE → PHVVDJH
- How can we crack this encrypted message:
  DEEDUSEKBTFEIIYRBOTUSETUJXYI

Caesar Cipher (cont’d)

```
DEEDUSEKBTFEIIYRBOTUSETUJXYI     QRRQHFRXOGSRVVLEOBGHFRGHWKLV
EFFEVTFLCUGFJJZSCPUVTFUVKYZJ     RSSRIGSYPHTSWWMFPCHIGSHIXLMW
FGGFWUGMDVHGKKATDQVWUGVWLZAK     STTSJHTZQIUTXXNGQDIJHTIJYMNX
GHHGXVHNEWIHLLBUERWXVHWXMABL     TUUTKIUARJVUYYOHREJKIUJKZNOY
HIIHYWIOFXJIMMCVFSXYWIXYNBCM     UVVULJVBSKWVZZPISFKLJVKLAOPZ
IJJIZXJPGYKJNNDWGTYZXJYZOCDN     VWWVMKWCTLXWAAQJTGLMKWLMBPQA
JKKJAYKQHZLKOOEXHUZAYKZAPDEO     WXXWNLXDUMYXBBRKUHMNLXMNCQRB
KLLKBZLRIAMLPPFYIVABZLABQEFP     XYYXOMYEVNZYCCSLVINOMYNODRSC
LMMLCAMSJBNMQQGZJWBCAMBCRFGQ     YZZYPNZFWOAZDDTMWJOPNZOPESTD
MNNMDBNTKCONRRHAKXCDBNCDSGHR     ZAAZQOAGXPBAEEUNXKPQOAPQFTUE
NOONECOULDPOSSIBLYDECODETHIS     ABBARPBHYQCBFFVOYLQRPBQRGUVF
OPPOFDPVMEQPTTJCMZEFDPEFUIJT     BCCBSQCIZRDCGGWPZMRSQCRSHVWG
PQQPGEQWNFRQUUKDNAFGEQFGVJKU     CDDCTRDJASEDHHXQANSTRDSTIWXH
```

- How long would it take a computer to try all 25 shifts?

Stream vs. Block Ciphers

- Caeser Cipher is also an example of a stream cipher: encodes one character at a time
- Block ciphers: a block (group) of symbols gets encoded into a block of cipher text.
  - Destroys the structure of the plaintext

Vigenère Cipher

- Shift different amount for each letter.

	ABCDEFGHIJKLMNOPQRSTUVWXYZ	
A	ABCDEFGHIJKLMNOPQRSTUVWXYZ	
B	BCDEFGHIJKLMNOPQRSTUVWXYZA	
C	CDEFGHIJKLMNOPQRSTUVWXYZAB	
D	DEFGHIJKLMNOPQRSTUVWXYZABC	
E	EFGHIJKLMNOPQRSTUVWXYZABCD	
F	FGHIJKLMNOPQRSTUVWXYZABCDE	etc.

	ABCDEFGHIJKLMNOPQRSTUVWXYZ
A	ABCDEFGHIJKLMNOPQRSTUVWXYZ
B	BCDEFGHIJKLMNOPQRSTUVWXYZA
C	CDEFGHIJKLMNOPQRSTUVWXYZAB
D	DEFGHIJKLMNOPQRSTUVWXYZABC
E	EFGHIJKLMNOPQRSTUVWXYZABCD
F	FGHIJKLMNOPQRSTUVWXYZABCDE

. . .

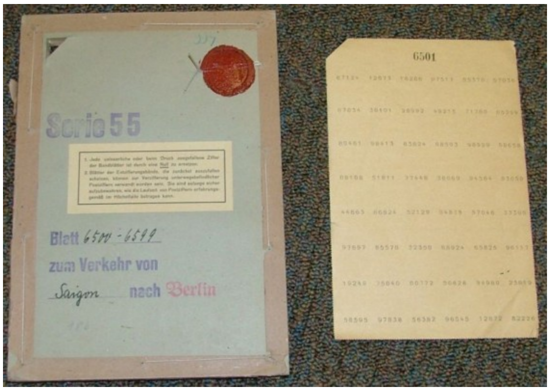
- Pick a secret key: DECAF
- Write Key over and over: DECAFDECAFDE
- Message: ATTACKATDAWN
- Encrypted: DXVAHNEVDFZR

Vernam Cipher

- Vigenère cipher was broken by Charles Babbage in the mid 1800s
  - The length of the key determines the cycle in which the cipher is repeated.
- Vernam Cipher: To encode a plaintext of *n* characters use a key of length *n*.

One-time Pads

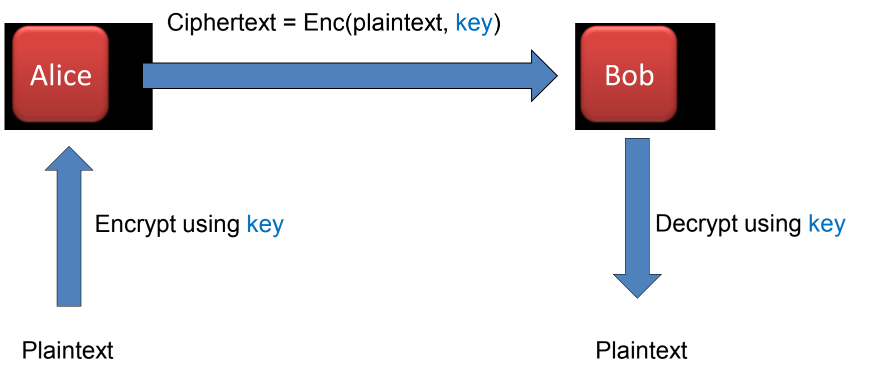
- Vernam cipher is commonly referred to as a one-time pad.



Alice and Bob have identical pads (shared keys)

- If random keys are used one-time pads are unbreakabke in theory.

Symmetric (Shared Key) Encryption



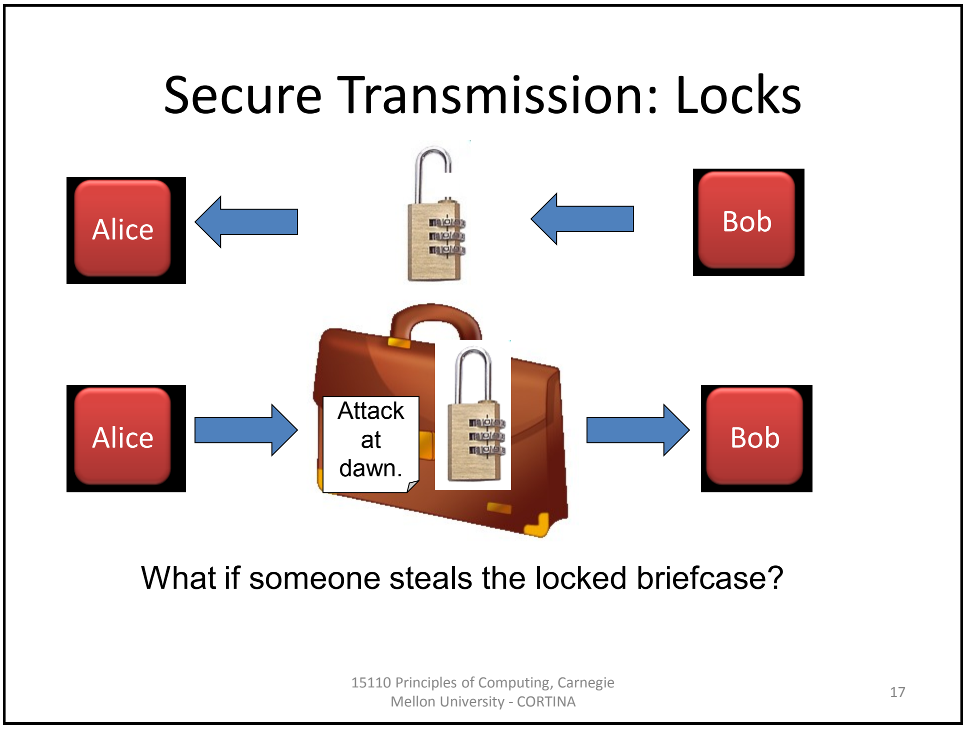
Alice uses the shared key to encrypt the plaintext to produce the ciphertext

Bob uses the shared key to decrypt the ciphertext to recover the plaintext

Shared Keys

- What makes a good key?
- How do you get the secret key to the receiver?

"Secure communication was practical only for people who could arrange to meet beforehand, or who had access to a prior method of secure communication (such as military couriers) for carrying the key between them. If Internet communications had to proceed on this assumption, electronic commerce never could have gotten off the ground.“
(from *Blown To Bits*)



Secure Transmission: Locks

Alice

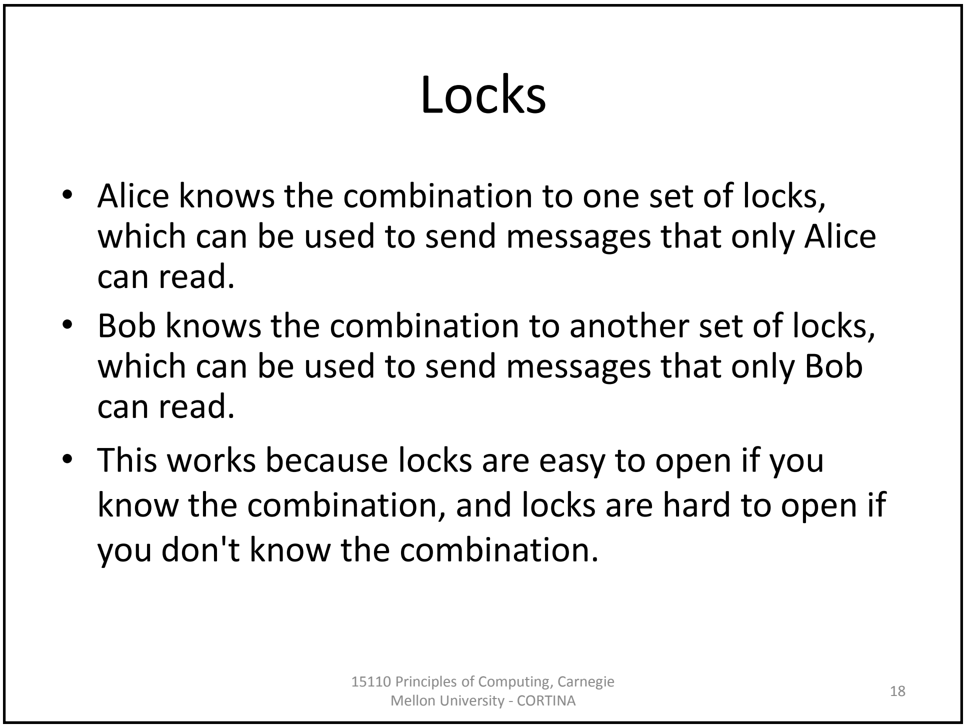
Bob

Alice

Attack
at
dawn.

Bob

What if someone steals the locked briefcase?



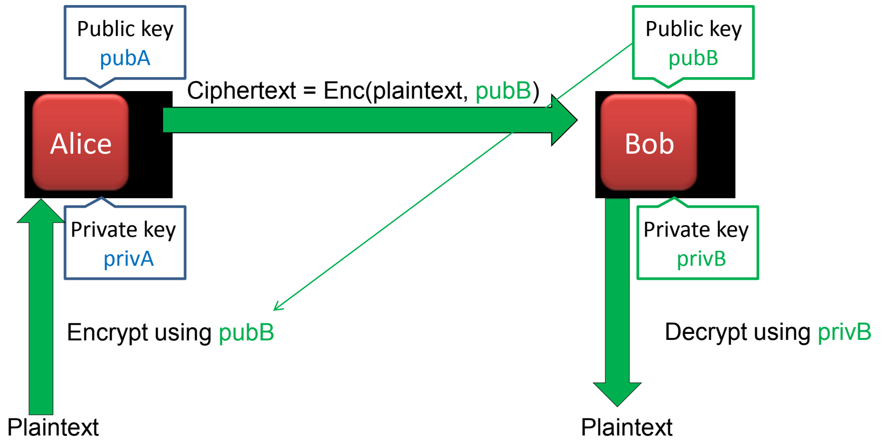
Locks

- Alice knows the combination to one set of locks, which can be used to send messages that only Alice can read.
- Bob knows the combination to another set of locks, which can be used to send messages that only Bob can read.
- This works because locks are easy to open if you know the combination, and locks are hard to open if you don't know the combination.

Locks

- How do you open a lock, if you don't know the combination?
  - If there are 3 digits, how many combinations do we need to try? (worst case)
- Suppose someone can crack my 3-digit combo lock in 15 minutes, by trying every combination. Do I give up on combo locks? No, I use more digits!
  - How long to crack a 6-digit lock at this rate? 10 days
  - How long to crack a 12-digit lock at this rate? 30,000 years
- Locks on the Internet: Public key encryption

Asymmetric (Public Key) Encryption



Public key
pubA

Ciphertext = Enc(plaintext, pubB)

Public key
pubB

Alice

Bob

Private key
privA

Private key
privB

Encrypt using pubB

Decrypt using privB

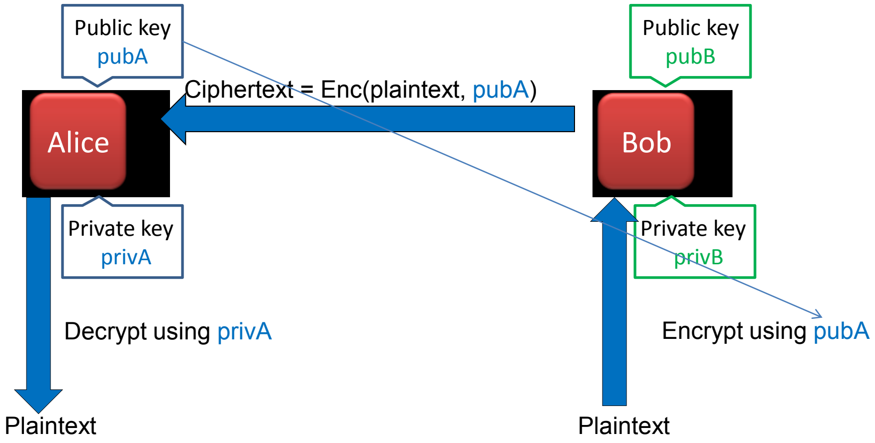
Plaintext

Plaintext

Alice uses Bob's public key to encrypt the plaintext to produce the ciphertext

Bob uses his private key to decrypt the ciphertext to recover the plaintext

Asymmetric (Public Key) Encryption



Alice uses her private key to decrypt the ciphertext to recover the plaintext

Bob uses Alice's public key To encrypt the plaintext to produce the ciphertext

RSA Encryption

- Current encryption technique for transmitting data on the Internet
  - Named after its inventors: Rivest, Shamir and Adleman
  - The URL at the top of the browser will begin with https://
  - The information you send using the HTTPS protocol is more secure than any encrypted military order sent during World War I, World War II, The Korean War, or The Vietnam War.

How RSA works

- First, we must be able to represent any message as a single number.
- For example:

  A T **T** A **C** K **A** T **D** A **W** N

  0120**20**01**03**11**01**20**04**01**23**14

Public and Private Keys

- Every receiver has a public key (e, n) and a private key (d, n).
- The transmitter encodes a (numerical) message M into an encrypted message C using the receiver’s public key:

$$M^{e} \text{ modulo } n \rightarrow C$$

- The receiver decodes the encrypted message C to get the original message M using the private key (which no one else knows).

$$C^{d} \text{ modulo } n \rightarrow M$$

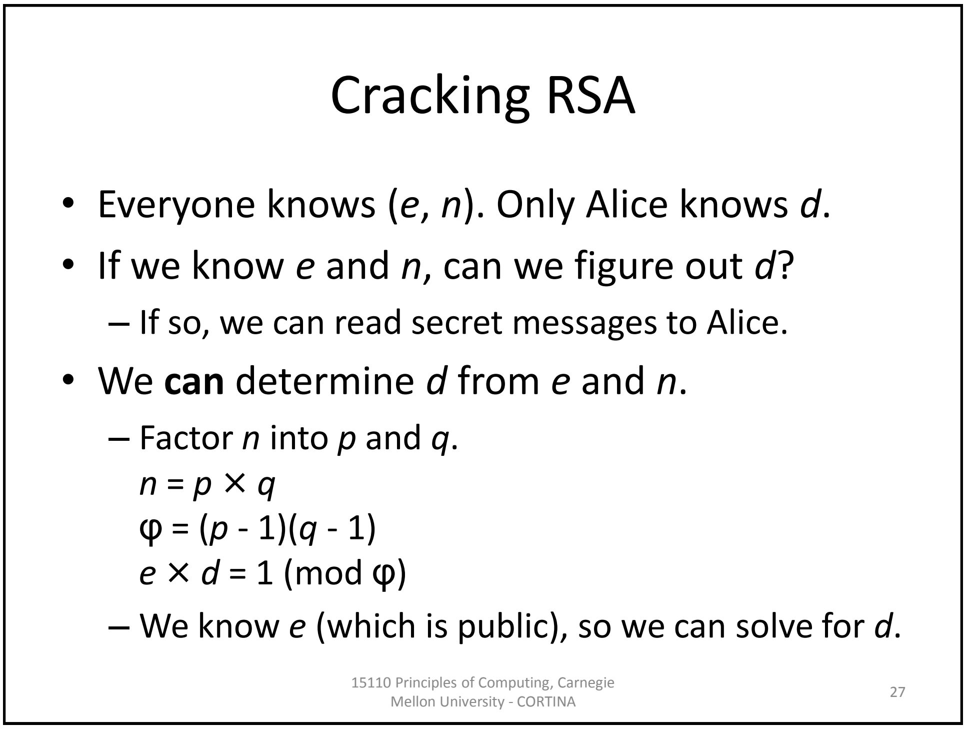
Example

- Alice’s Public Key: (3, 33) (e = 3, n = 33)
- Alice’s Private Key: (7, 33) (d = 7, n = 33)
  - Usually these are really huge numbers with many hundreds of digits!
- Bob wants to send the message 4
  - Bob encrypts the message using e and n:
    4^3 modulo 33 ⟶ 31 ... Bob sends 31
- Alice receives the encoded message 31
  - Alice decrypts the message using d and n:
    31^7 modulo 33 ⟶ 4

Simple Example: Computing e, n and d

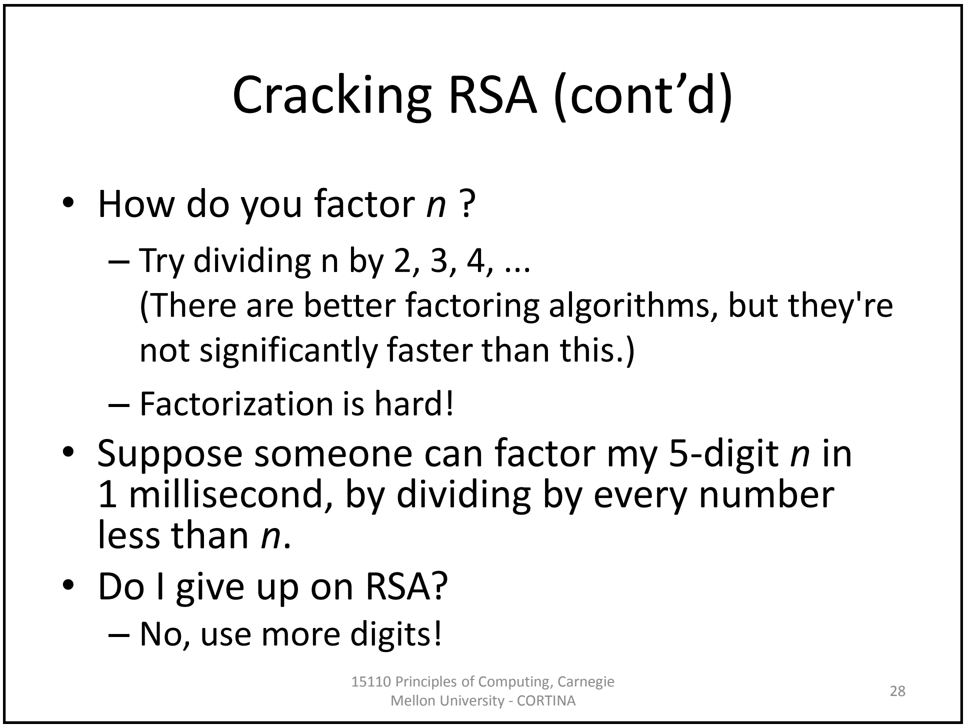
• p and q are (big) random primes.	$p = 3$, $q = 11$
• $n = p \times q$	$n = 3 \times 11 = 33$
• $\varphi = (p - 1)(q - 1)$	$\varphi = 2 \times 10 = 20$
• e is small and relatively prime to φ	$e = 3$
• d, such that: $e \times d \bmod \varphi = 1$	$3 \times d \bmod 20 = 1$ $d = 7$

Usually the primes are huge numbers--hundreds of digits long.



Cracking RSA

- Everyone knows (*e*, *n*). Only Alice knows *d*.
- If we know *e* and *n*, can we figure out *d*?
  - If so, we can read secret messages to Alice.
- We **can** determine *d* from *e* and *n*.
  - Factor *n* into *p* and *q*.
    $n = p \times q$
    $\varphi = (p - 1)(q - 1)$
    $e \times d = 1 \pmod{\varphi}$
  - We know *e* (which is public), so we can solve for *d*.



Cracking RSA (cont'd)

- How do you factor *n* ?
  - Try dividing n by 2, 3, 4, ...
    (There are better factoring algorithms, but they're not significantly faster than this.)
  - Factorization is hard!
- Suppose someone can factor my 5-digit *n* in 1 millisecond, by dividing by every number less than *n*.
- Do I give up on RSA?
  - No, use more digits!

RSA is safe (for now)

- Suppose someone can factor my 5-digit *n* in 1 ms, by dividing by every number less than *n*.
- At this rate, to factor a 10-digit number would take 2 minutes.
- At this rate, to factor a 15-digit number would take 4 months.
- At this rate, to factor a 20-digit number would take 30,000 years.
- At this rate, to factor a 25-digit number would take 3 billion years.
- We're safe with RSA!

One Way Functions

- One way functions are easy to compute but hard to invert.
- Calculating y = f(x) is easy, but finding the value of x given y is hard.
- Computing the product of two prime numbers is a one way function because factoring is hard.

One Way Hash Functions

- Given a message M, compute a hash or “digest” function D that is shorter than M but still long enough to prohibit exhaustive search through the space of hash values, e.g., 512 bytes.
  - Example hashes: MD5 (cracked), SHA-1 (not perfect).
- If the digest function is properly designed, you will not be able to find another message M’ that has the same digest D as M.
- When publishing a large file, also publish the digest D so anyone can check for corruption.

Digital Signatures

- How can Alice prove that she is the author of a message M?
  – Compute the digest D of M.
  – Encrypt the digest using her private key, giving the signature D’.
  – Publish both M and the signature D’.
- Anyone who reads M can compute D. They can also decrypt D’ using Alice’s public key to verify that it matches D.

Certificate Authorities

- How do we know that “Alice” is really Alice? (Or that “Microsoft” is really Microsoft?)
- *Certificate Authorities* sign digital certificates indicating authenticity of a sender who they have checked out in the real world.
- Senders provide copies of their certificates along with their message or software.
- Verisign: major certificate authority in the US.
- But can we trust the certificate authorities?